# CompSci 102
# Discrete Math for Computer Science

DEZA RWZMLW HLCXTYR

February 28, 2012

Prof. Rodger

# Announcements

- Prof. Rodger office hours this week
  – Only Tuesday 1-1:30pm
- Read for next time Chap. 5.1
- Exam 1 back today
- No Recitation this week

# Chap 4.4 - Linear Congruences

**Definition**: A congruence of the form

$$ax \equiv b(\text{ mod } m),$$

where $m$ is a positive integer, $a$ and $b$ are integers, and $x$ is a variable, is called a *linear congruence*.

- The solutions to a linear congruence $ax \equiv b(\text{ mod } m)$ are all integers $x$ that satisfy the congruence.

**Definition**: An integer $\bar{a}$ such that $\bar{a}a \equiv 1(\text{ mod } m)$ is said to be an *inverse* of $a$ modulo $m$.

**Example**: What is the inverse of 3 modulo 7?

- One method of solving linear congruences makes use of an inverse $\bar{a}$, if it exists. Although we can not divide both sides of the congruence by $a$, we can multiply by $\bar{a}$ to solve for $x$.

# Inverse of *a* modulo *m*

- The following theorem guarantees that an inverse of $a$ modulo $m$ exists whenever $a$ and $m$ are relatively prime. Two integers $a$ and $b$ are relatively prime when $\gcd(a,b) = 1$.

**Theorem 1**: If $a$ and $m$ are relatively prime integers and $m > 1$, then an inverse of $a$ modulo $m$ exists. Furthermore, this inverse is unique modulo $m$. (This means that there is a unique positive integer $\bar{a}$ less than $m$ that is an inverse of $a$ modulo $m$ and every other inverse of $a$ modulo $m$ is congruent to $\bar{a}$ modulo $m$.)

**Proof**: Since $\gcd(a,m) = 1$, by Theorem 6 of Section 4.3, there are integers $s$ and $t$ such that $sa + tm = 1$.

# Finding Inverses

- The Euclidean algorithm and Bézout coefficients gives us a systematic approaches to finding inverses.

**Example**: Find an inverse of 3 modulo 7.

**Solution**: Because gcd(3,7) = 1, by Theorem 1, an inverse of 3 modulo 7 exists.

- Using the Euclidian algorithm to find gcd: $7 = 2\cdot3 + 1$.
- From this equation, we get $-2\cdot3 + 1\cdot7 = 1$, and see that $-2$ and 1 are Bézout coefficients of 3 and 7.
- Hence, $-2$ is an inverse of 3 modulo 7.
- Also every integer congruent to $-2$ modulo 7 is an inverse of 3 modulo 7, i.e., 5, $-9$, 12, etc.

# Finding Inverses

**Example**: Find an inverse of 101 modulo 4620.

**Solution**: First use the Euclidian algorithm to show that gcd(101,4620) = 1.

Working Backwards:

# Using Inverses to Solve Congruences

- We can solve the congruence $ax \equiv b(\bmod m)$ by multiplying both sides by $\bar{a}$.

**Example**: What are the solutions of the congruence $3x \equiv 4(\bmod 7)$.

# The Chinese Remainder Theorem

- In the first century, the Chinese mathematician Sun-Tsu asked:

  There are certain things whose number is unknown. When divided by 3, the remainder is 2; when divided by 5, the remainder is 3; when divided by 7, the remainder is 2. What will be the number of things?

- This puzzle can be translated into the solution of the system of congruences:

  $x \equiv 2 (\bmod 3)$,
  $x \equiv 3 (\bmod 5)$,
  $x \equiv 2 (\bmod 7)$?

- We'll see how the theorem that is known as the *Chinese Remainder Theorem* can be used to solve Sun-Tsu's problem.

# The Chinese Remainder Theorem

**Theorem 2**: (*The Chinese Remainder Theorem*) Let $m_1,m_2,\ldots,m_n$ be pairwise relatively prime positive integers greater than one and $a_1,a_2,\ldots,a_n$ arbitrary integers. Then the system

$$\begin{aligned} x &\equiv a_1 \pmod{m_1} \\ x &\equiv a_2 \pmod{m_2} \\ &\cdot \\ &\cdot \\ &\cdot \\ x &\equiv a_n \pmod{m_n} \end{aligned}$$

has a unique solution modulo $m = m_1m_2 \cdots m_n$.

(That is, there is a solution x with $0 \leq x < m$ and all other solutions are congruent modulo $m$ to this solution.)

- **Proof**: We'll show that a solution exists by describing a way to construct the solution. Showing that the solution is unique modulo $m$ is Exercise 30.

*continued* →

# The Chinese Remainder Theorem

To construct a solution first let $M_k = m/m_k$ for $k = 1,2,\ldots,n$ and $m = m_1m_2 \cdots m_n$.

Since $\gcd(m_k, M_k) = 1$, by Theorem 1, there is an integer $y_k$, an inverse of $M_k$ modulo $m_k$, such that

$$M_k\, y_k \equiv 1 \pmod{m_k}.$$

Form the sum

$$x = a_1 M_1 y_1 + a_2 M_2 y_2 + \cdots + a_n M_n y_n .$$

Note that because $M_j \equiv 0 \pmod{m_k}$ whenever $j \neq k$, all terms except the $k$th term in this sum are congruent to 0 modulo $m_k$.

Because $M_k\, y_k \equiv 1 \pmod{m_k}$, we see that $x \equiv a_k M_k\, y_k \equiv a_k \pmod{m_k}$, for $k = 1,2,\ldots,n$.

Hence, $x$ is a simultaneous solution to the $n$ congruences.

$$\begin{aligned} x &\equiv a_1 \pmod{m_1} \\ x &\equiv a_2 \pmod{m_2} \\ &\cdot \\ &\cdot \\ &\cdot \\ x &\equiv a_n \pmod{m_n} \end{aligned}$$

# The Chinese Remainder Theorem

**Example**: Consider the 3 congruences from Sun-Tsu's problem:

$x \equiv 2 \pmod 3$, $x \equiv 3 \pmod 5$, $x \equiv 2 \pmod 7$.

- Let $m = 3 \cdot 5 \cdot 7 = 105$, $M_1 = m/3 = 35$, $M_3 = m/5 = 21$, $M_3 = m/7 = 15$.
- We see that
- Hence,

# Back Substitution

- We can also solve systems of linear congruences with pairwise relatively prime moduli by rewriting a congruences as an equality using Theorem 4 in Section 4.1, substituting the value for the variable into another congruence, and continuing the process until we have worked through all the congruences. This method is known as *back substitution*.

**Example**: Use the method of back substitution to find all integers $x$ such that $x \equiv 1 \pmod 5$, $x \equiv 2 \pmod 6$, and $x \equiv 3 \pmod 7$.

**Solution**: By Theorem 4 in Section 4.1, the first congruence can be rewritten as $x = 5t + 1$, where $t$ is an integer.

- Substituting into the second congruence yields $5t + 1 \equiv 2 \pmod 6$.
- Solving this tells us that $t \equiv 5 \pmod 6$.
- Using Theorem 4 again gives $t = 6u + 5$ where $u$ is an integer.
- Substituting this back into $x = 5t + 1$, gives $x = 5(6u + 5) + 1 = 30u + 26$.
- Inserting this into the third equation gives $30u + 26 \equiv 3 \pmod 7$.
- Solving this congruence tells us that $u \equiv 6 \pmod 7$.
- By Theorem 4, $u = 7v + 6$, where $v$ is an integer.
- Substituting this expression for $u$ into $x = 30u + 26$, tells us that $x = 30(7v + 6) + 26 = 210u + 206$.

Translating this back into a congruence we find the solution $x \equiv 206 \pmod{210}$.

# Fermat’s Little Theorem

Pierre de Fermat
(1601-1665)

**Theorem 3**: (*Fermat’s Little The*orem) If $p$ is prime and $a$ is an integer not divisible by $p$, then $a^{p-1} \equiv 1 \pmod{p}$

Furthermore, for every integer $a$ we have $a^p \equiv a \pmod{p}$

(*proof outlined in Exercise 19*)

Fermat’s little theorem is useful in computing the remainders modulo $p$ of large powers of integers.

**Example**: Find $7^{222}$ **mod** 11.

By Fermat’s little theorem, we know that $7^{10} \equiv 1 \pmod{11}$, and so $(7^{10})^k \equiv 1 \pmod{11}$, for every positive integer $k$. Therefore,

# Pseudoprimes

- By Fermat’s little theorem $n > 2$ is prime, where $2^{n-1} \equiv 1 \pmod{n}$.
- But if this congruence holds, $n$ may not be prime. Composite integers $n$ such that $2^{n-1} \equiv 1 \pmod{n}$ are called *pseudoprimes* to the base 2.

**Example**: The integer 341 is a pseudoprime to the base 2.

- We can replace 2 by any integer $b \geq 2$.

**Definition**: Let $b$ be a positive integer. If $n$ is a composite integer, and $b^{n-1} \equiv 1 \pmod{n}$, then $n$ is called a *pseudoprime to the base b*.

# Pseudoprimes

- Given a positive integer $n$, such that $2^{n-1} \equiv 1 \pmod{n}$:
  - If $n$ does not satisfy the congruence, it is composite.
  - If $n$ does satisfy the congruence, it is either prime or a pseudoprime to the base 2.
- Doing similar tests with additional bases $b$, provides more evidence as to whether $n$ is prime.
- Among the positive integers not exceeding a positive real number $x$, compared to primes, there are relatively few pseudoprimes to the base $b$.
  - For example, among the positive integers less than $10^{10}$ there are 455,052,512 primes, but only 14,884 pseudoprimes to the base 2.

# Primitive Roots

**Definition**: A primitive root modulo a prime $p$ is an integer $r$ in $\mathbf{Z}_p$ such that every nonzero element of $\mathbf{Z}_p$ is a power of $r$.

**Example**: Since every element of $\mathbf{Z}_{11}$ is a power of 2, 2 is a primitive root of 11.

**Example**: Since not all elements of $\mathbf{Z}_{11}$ are powers of 3, 3 is not a primitive root of 11.

**Important Fact**: There is a primitive root modulo $p$ for every prime number $p$.

# Discrete Logarithms

Suppose $p$ is prime and $r$ is a primitive root modulo $p$. If $a$ is an integer between 1 and $p - 1$, that is an element of $\mathbf{Z}_p$, there is a unique exponent $e$ such that $r^e = a$ in $\mathbf{Z}_p$, that is, $r^e \bmod p = a$.

**Definition**: Suppose that $p$ is prime, $r$ is a primitive root modulo $p$, and $a$ is an integer between 1 and $p - 1$, inclusive. If $r^e \bmod p = a$ and $1 \leq e \leq p - 1$, we say that $e$ is the *discrete logarithm* of $a$ modulo $p$ to the base $r$ and we write $\log_r a = e$ (where the prime $p$ is understood).

**Example 1**: We write $\log_2 3 = 8$ since the discrete logarithm of 3 modulo 11 to the base 2 is 8 as $2^8 = 3$ modulo 11.

**Example 2**: We write $\log_2 5 = 4$ since the discrete logarithm of 5 modulo 11 to the base 2 is 4 as $2^4 = 5$ modulo 11.

There is no known polynomial time algorithm for computing the discrete logarithm of $a$ modulo $p$ to the base $r$ (when given the prime $p$, a root $r$ modulo $p$, and a positive integer $a \in \mathbf{Z}_p$). The problem plays a role in cryptography as will be discussed in Section 4.6.

# Chap 4.5 - Hashing Functions

**Definition**: A *hashing function h* assigns memory location $h(k)$ to the record that has $k$ as its key.

- A common hashing function is $h(k) = k$ **mod** $m$, where $m$ is the number of memory locations.
- Because this hashing function is onto, all memory locations are possible.

**Example**: Let $h(k) = k$ **mod** 111. This hashing function assigns the records of customers with social security numbers as keys to memory locations in the following manner:

h(064212848) = 064212848 **mod** 111 = 14

h(037149212) = 037149212 **mod** 111 = 65

h(107405723) = 107405723 **mod** 111 = 14, but since location 14 is already occupied, the record is assigned to the next available position, which is 15.

# More on Hashing Functions

- The hashing function is not one-to-one as there are many more possible keys than memory locations. When more than one record is assigned to the same location, we say a *collision* occurs. Here a collision has been resolved by assigning the record to the first free location.
- For collision resolution, we can use a *linear probing function*:

  $h(k,i) = (h(k) + i)$ **mod** $m$, where $i$ runs from 0 to $m - 1$.
- There are many other methods of handling with collisions.

# Pseudorandom Numbers

- Randomly chosen numbers are needed for many purposes, including computer simulations.
- *Pseudorandom numbers* are not truly random since they are generated by systematic methods.
- The *linear congruential method* is one commonly used procedure for generating pseudorandom numbers.
- Four integers are needed: the *modulus m*, the *multiplier a*, the *increment c*, and *seed* $x_0$, with $2 \leq a < m$, $0 \leq c < m$, $0 \leq x_0 < m$.
- We generate a sequence of pseudorandom numbers $\{x_n\}$, with $0 \leq x_n < m$ for all n, by successively using the recursively defined function

$$x_{n+1} = (ax_n + c) \textbf{ mod } m.$$

(*an example of a recursive definition, discussed in Section* 5.3)

- If pseudorandom numbers between 0 and 1 are needed, then the generated numbers are divided by the modulus, $x_n / m$.

# Pseudorandom Numbers

- **Example**: Find the sequence of pseudorandom numbers generated by the linear congruential method with modulus $m = 9$, multiplier $a = 7$, increment $c = 4$, and seed $x_0 = 3$.
- **Solution**: Compute the terms of the sequence by successively using the congruence $x_{n+1} = (7x_n + 4)$ **mod** 9, with $x_0 = 3$.

$x_1 = 7x_0 + 4 \textbf{ mod } 9 = 7\cdot 3 + 4 \textbf{ mod } 9 = 25 \textbf{ mod } 9 = 7,$
$x_2 = 7x_1 + 4 \textbf{ mod } 9 = 7\cdot 7 + 4 \textbf{ mod } 9 = 53 \textbf{ mod } 9 = 8,$
$x_3 = 7x_2 + 4 \textbf{ mod } 9 = 7\cdot 8 + 4 \textbf{ mod } 9 = 60 \textbf{ mod } 9 = 6,$
$x_4 = 7x_3 + 4 \textbf{ mod } 9 = 7\cdot 6 + 4 \textbf{ mod } 9 = 46 \textbf{ mod } 9 = 1,$
$x_5 = 7x_4 + 4 \textbf{ mod } 9 = 7\cdot 1 + 4 \textbf{ mod } 9 = 11 \textbf{ mod } 9 = 2,$
$x_6 = 7x_5 + 4 \textbf{ mod } 9 = 7\cdot 2 + 4 \textbf{ mod } 9 = 18 \textbf{ mod } 9 = 0,$
$x_7 = 7x_6 + 4 \textbf{ mod } 9 = 7\cdot 0 + 4 \textbf{ mod } 9 = 4 \textbf{ mod } 9 = 4,$
$x_8 = 7x_7 + 4 \textbf{ mod } 9 = 7\cdot 4 + 4 \textbf{ mod } 9 = 32 \textbf{ mod } 9 = 5,$
$x_9 = 7x_8 + 4 \textbf{ mod } 9 = 7\cdot 5 + 4 \textbf{ mod } 9 = 39 \textbf{ mod } 9 = 3.$

The sequence generated is 3,7,8,6,1,2,0,4,5,3,7,8,6,1,2,0,4,5,3,...

It repeats after generating 9 terms.

- Commonly, computers use a linear congruential generator with increment $c = 0$. This is called a *pure multiplicative generator*. Such a generator with modulus $2^{31} - 1$ and multiplier $7^5 = 16{,}807$ generates $2^{31} - 2$ numbers before repeating.

# Check Digits: UPCs

- A common method of detecting errors in strings of digits is to add an extra digit at the end, which is evaluated using a function. If the final digit is not correct, then the string is assumed not to be correct.

**Example**: Retail products are identified by their *Universal Product Codes* (*UPC*s). Usually these have 12 decimal digits, the last one being the check digit. The check digit is determined by the congruence:

$$3x_1 + x_2 + 3x_3 + x_4 + 3x_5 + x_6 + 3x_7 + x_8 + 3x_9 + x_{10} + 3x_{11} + x_{12} \equiv 0 \pmod{10}.$$

a. Suppose that the first 11 digits of the UPC are 79357343104. What is the check digit?

b. Is 041331021641 a valid UPC?

**Solution**:

a. $3\cdot 7 + 9 + 3\cdot 3 + 5 + 3\cdot 7 + 3 + 3\cdot 4 + 3 + 3\cdot 1 + 0 + 3\cdot 4 + x_{12} \equiv 0 \pmod{10}$
$21 + 9 + 9 + 5 + 21 + 3 + 12 + 3 + 3 + 0 + 12 + x_{12} \equiv 0 \pmod{10}$
$98 + x_{12} \equiv 0 \pmod{10}$
$x_{12} \equiv 0 \pmod{10}$ So, the check digit is 2.

b. $3\cdot 0 + 4 + 3\cdot 1 + 3 + 3\cdot 3 + 1 + 3\cdot 0 + 2 + 3\cdot 1 + 6 + 3\cdot 4 + 1 \equiv 0 \pmod{10}$
$0 + 4 + 3 + 3 + 9 + 1 + 0 + 2 + 3 + 6 + 12 + 1 = 44 \equiv 4 \not\equiv \pmod{10}$
Hence, 041331021641 is not a valid UPC.

# Check Digits:ISBNs

**B**ooks are identified by an *International Standard Book Number* (ISBN-10), a 10 digit code. The first 9 digits identify the language, the publisher, and the book. The tenth digit is a check digit, which is determined by the following congruence

$$x_{10} \equiv \sum_{i=1}^{9} i x_i \pmod{11}.$$

The validity of an ISBN-10 number can be evaluated with the equivalent $\sum_{i=1}^{10} i x_i \equiv 0 \pmod{11}.$

a. Suppose that the first 9 digits of the ISBN-10 are 007288008. What is the check digit?

b. Is 084930149X a valid ISBN10?

**Solution**:

a. $X_{10} \equiv 1\cdot 0 + 2\cdot 0 + 3\cdot 7 + 4\cdot 2 + 5\cdot 8 + 6\cdot 8 + 7\cdot 0 + 8\cdot 0 + 9\cdot 8 \pmod{11}.$
$X_{10} \equiv 0 + 0 + 21 + 8 + 40 + 48 + 0 + 0 + 72 \pmod{11}.$
$X_{10} \equiv 189 \equiv 2 \pmod{11}$. Hence, $X_{10} = 2$.

b. $1\cdot 0 + 2\cdot 8 + 3\cdot 4 + 4\cdot 9 + 5\cdot 3 + 6\cdot 0 + 7\cdot 1 + 8\cdot 4 + 9\cdot 9 + 10\cdot 10 =$
$0 + 16 + 12 + 36 + 15 + 0 + 7 + 32 + 81 + 100 = 299 \equiv 2 \not\equiv 0 \pmod{11}$
Hence, 084930149X is not a valid ISBN-10.

X is used for the digit 10.

- A *single error* is an error in one digit of an identification number and a *transposition error* is the accidental interchanging of two digits. Both of these kinds of errors can be detected by the check digit for ISBN-10. (*see text for more details*)

# Chap. 4.6 - Caesar Cipher

Julius Caesar created secret messages by shifting each letter three letters forward in the alphabet (sending the last three letters to the first three letters.) For example, the letter B is replaced by E and the letter X is replaced by A. This process of making a message secret is an example of *encryption*.

Here is how the encryption process works:

- Replace each letter by an integer from $\mathbf{Z}_{26}$, that is an integer from 0 to 25 representing one less than its position in the alphabet.
- The encryption function is $f(p) = (p + 3)$ **mod** 26. It replaces each integer $p$ in the set $\{0,1,2,\ldots,25\}$ by $f(p)$ in the set $\{0,1,2,\ldots,25\}$ .
- Replace each integer $p$ by the letter with the position $p + 1$ in the alphabet.

**Example**: Encrypt the message "MEET YOU IN THE PARK" using the Caesar cipher.

**Solution**: 12 4 4 19 24 14 20 8 13 19 7 4 15 0 17 10.

Now replace each of these numbers $p$ by $f(p) = (p + 3)$ **mod** 26.

15 7 7 22 1 17 23 11 16 22 10 7 18 3 20 13.

Translating the numbers back to letters produces the encrypted message

"PHHW BRX LQ WKH SDUN."

# Caesar Cipher

- To recover the original message, use $f^{-1}(p) = (p-3)$ **mod** 26. So, each letter in the coded message is shifted back three letters in the alphabet, with the first three letters sent to the last three letters. This process of recovering the original message from the encrypted message is called *decryption*.
- The Caesar cipher is one of a family of ciphers called *shift ciphers*. Letters can be shifted by an integer $k$, with 3 being just one possibility. The encryption function is

  $f(p) = (p + k)$ **mod** 26

  and the decryption function is

  $f^{-1}(p) = (p-k)$ **mod** 26

  The integer $k$ is called a *key*.

# Shift Cipher

**Example 1**: Encrypt the message "STOP GLOBAL WARMING" using the shift cipher with $k = 11$.

**Solution**: Replace each letter with the corresponding element of $\mathbf{Z}_{26}$.

# Shift Cipher

**Example 2**: Decrypt the message "LEWLYPLUJL PZ H NYLHA ALHJOLY" that was encrypted using the shift cipher with $k = 7$.

**Solution**: Replace each letter with the corresponding element of $\mathbf{Z}_{26}$.

# Affine Ciphers

- Shift ciphers are a special case of *affine ciphers* which use functions of the form

  $f(p) = (ap + b)$ **mod** 26,

  where $a$ and $b$ are integers, chosen so that $f$ is a bijection.

  The function is a bijection if and only if $\gcd(a,26) = 1$.
- **Example**: What letter replaces the letter K when the function $f(p) = (7p + 3)$ **mod** 26 is used for encryption.

  **Solution**:
- To decrypt a message encrypted by a shift cipher, the congruence $c \equiv ap + b \pmod{26}$ needs to be solved for $p$.
  - Subtract $b$ from both sides to obtain $c - b \equiv ap \pmod{26}$.
  - Multiply both sides by the inverse of $a$ modulo 26, which exists since $\gcd(a,26) = 1$.
  - $\bar{a}(c - b) \equiv \bar{a}ap \pmod{26}$, which simplifies to $\bar{a}(c - b) \equiv p \pmod{26}$.
  - $p \equiv \bar{a}(c - b) \pmod{26}$ is used to determine $p$ in $\mathbf{Z}_{26}$.

# Cryptanalysis of Affine Ciphers

- The process of recovering plaintext from ciphertext without knowledge both of the encryption method and the key is known as *cryptanalysis* or *breaking codes*.
- An important tool for cryptanalyzing ciphertext produced with a affine ciphers is the relative frequencies of letters. The nine most common letters in the English texts are E 13%, T 9%, A 8%, O 8%, I 7%, N 7%, S 7%, H 6%, and R 6%.
- To analyze ciphertext:
  - Find the frequency of the letters in the ciphertext.
  - Hypothesize that the most frequent letter is produced by encrypting E.
  - If the value of the shift from E to the most frequent letter is *k*, shift the ciphertext by $-k$ and see if it makes sense.
  - If not, try T as a hypothesis and continue.

- **Example**: We intercepted the message “ZNK KGXRE HOXJ MKZY ZNK CUXS” that we know was produced by a shift cipher. Let’s try to cryptanalyze.
- **Solution**:

# Block Ciphers

- Ciphers that replace each letter of the alphabet by another letter are called *character* or *monoalphabetic* ciphers.
- They are vulnerable to cryptanalysis based on letter frequency. *Block ciphers* avoid this problem, by replacing blocks of letters with other blocks of letters.
- A simple type of block cipher is called the *transposition cipher*. The key is a permutation σ of the set $\{1,2,\ldots,m\}$, where *m* is an integer, that is a one-to-one function from $\{1,2,\ldots,m\}$ to itself.
- To encrypt a message, split the letters into blocks of size *m*, adding additional letters to fill out the final block. We encrypt $p_1,p_2,\ldots,p_m$ as $c_1,c_2,\ldots,c_m = p_{\sigma(1)},p_{\sigma(2)},\ldots,p_{\sigma(m)}$.
- To decrypt the $c_1,c_2,\ldots,c_m$ transpose the letters using the inverse permutation $\sigma^{-1}$.

# Block Ciphers

**Example**: Using the transposition cipher based on the permutation σ of the set $\{1,2,3,4\}$ with $\sigma(1) = 3$, $\sigma(2) = 1$, $\sigma(3) = 4$, $\sigma(4) = 2$,

a. Encrypt the plaintext PIRATE ATTACK
b. Decrypt the ciphertext message SWUE TRAEOEHS, which was encryted using the same cipher.

**Solution**:

# Cryptosystems

**Definition**: A *cryptosystem* is a five-tuple $(\mathcal{P},\mathcal{C},\mathcal{K},\mathcal{E},\mathcal{D})$, where

- $\mathcal{P}$ is the set of plainntext strings,
- $\mathcal{C}$ is the set of ciphertext strings,
- $\mathcal{K}$ is the *keyspace* (set of all possible keys),
- $\mathcal{E}$ is the set of encription functions, and
- $\mathcal{D}$ is the set of decryption functions.

- The encryption function in $\mathcal{E}$ corresponding to the key *k* is denoted by $E_k$ and the decription function in $\mathcal{D}$ that decrypts cipher text enrypted using $E_k$ is denoted by $D_k$. Therefore:

$$D_k(E_k(p)) = p, \text{ for all plaintext strings } p.$$

# Cryptosystems

**Example**: Describe the family of shift ciphers as a cryptosystem.

**Solution**: Assume the messages are strings consisting of elements in $\mathbf{Z}_{26}$.

# Public Key Cryptography

- All classical ciphers, including shift and affine ciphers, are *private key cryptosystems*. Knowing the encryption key allows one to quickly determine the decryption key.
- All parties who wish to communicate using a private key cryptosystem must share the key and keep it a secret.
- In public key cryptosystems, first invented in the 1970s, knowing how to encrypt a message does not help one to decrypt the message. Therefore, everyone can have a publicly known encryption key. The only key that needs to be kept secret is the decryption key.

# The RSA Cryptosystem

Clifford Cocks
(Born 1950)

- A public key cryptosystem, now known as the RSA system was introduced in 1976 by three researchers at MIT.

Ronald Rivest
(Born 1948)

Adi Shamir
(Born 1952)

Leonard
Adelman
(Born 1945)

- It is now known that the method was discovered earlier by Clifford Cocks, working secretly for the UK government.
- The public encryption key is $(n,e)$, where $n = pq$ (the modulus) is the product of two large (200 digits) primes $p$ and $q$, and an exponent $e$ that is relatively prime to $(p-1)(q-1)$. The two large primes can be quickly found using probabilistic primality tests, discussed earlier. But $n = pq$, with approximately 400 digits, cannot be factored in a reasonable length of time.

# RSA Encryption

- To encrypt a message using RSA using a key $(n,e)$ :

i. Translate the plaintext message $M$ into sequences of two digit integers representing the letters. Use 00 for A, 01 for B, etc.

ii. Concatenate the two digit integers into strings of digits.

iii. Divide this string into equally sized blocks of $2N$ digits where $2N$ is the largest even number 2525...25 with $2N$ digits that does not exceed $n$.

iv. The plaintext message M is now a sequence of integers $m_1, m_2, \ldots, m_k$.

v. Each block (an integer) is encrypted using the function $C = M^e \textbf{ mod } n$.

# RSA Encryption Example

**Example**: Encrypt the message STOP using the RSA cryptosystem with key(2537,13).

- $2537 = 43 \cdot 59$,
- $p = 43$ and $q = 59$ are primes and $\gcd(e,(p-1)(q-1)) = \gcd(13, 42 \cdot 58) = 1$.

**Solution**: Translate the letters in STOP to their numerical equivalents 18 19 14 15.

# RSA Decryption

- To decrypt a RSA ciphertext message, the decryption key $d$, an inverse of $e$ modulo $(p-1)(q-1)$ is needed. The inverse exists since $\gcd(e,(p-1)(q-1)) = \gcd(13, 42 \cdot 58) = 1$.
- With the decryption key $d$, we can decrypt each block with the computation $M = C^d \textbf{ mod } p \cdot q$. (*see text for full derivation*)
- RSA works as a public key system since the only known method of finding $d$ is based on a factorization of $n$ into primes. There is currently no known feasible method for factoring large numbers into primes.

# RSA Decryption

- **Example**: The message 0981 0461 is received. What is the decrypted message if it was encrypted using the RSA cipher from the previous example.

  **Solution**: The message was encrypted with $n = 43 \cdot 59$ and exponent 13. An inverse of 13 modulo $42 \cdot 58 = 2436$ (*exercise* 2 *in Section* 4.4) is $d = 937$.

# Cryptographic Protocols: Key Exchange

- *Cryptographic protocols* are exchanges of messages carried out by two or more parties to achieve a particular security goal.
- *Key exchange* is a protocol by which two parties can exchange a secret key over an insecure channel without having any past shared secret information. Here the *Diffe-Hellman key agreement protcol* is described by example.
  1. Suppose that Alice and Bob want to share a common key.
  2. Alice and Bob agree to use a prime $p$ and a primitive root $a$ of $p$.
  3. Alice chooses a secret integer $k_1$ and sends $a^{k_1} \textbf{ mod } p$ to Bob.
  4. Bob chooses a secret integer $k_2$ and sends $a^{k_2} \textbf{ mod } p$ to Alice.
  5. Alice computes $(a^{k_2})^{k_1} \textbf{ mod } p$.
  6. Bob computes $(a^{k_1})^{k_2} \textbf{ mod } p$.

  At the end of the protocol, Alice and Bob have their shared key

  $$(a^{k2})^{k1} \textbf{ mod } p = (a^{k_1})^{k_2} \textbf{ mod } p.$$
- To find the secret information from the public information would require the adversary to find $k_1$ and $k_2$ from $a^{k_1} \textbf{ mod } p$ and $a^{k_2} \textbf{ mod } p$ respectively. This is an instance of the discrete logarithm problem, considered to be computationally infeasible when $p$ and $a$ are sufficiently large.

# Cryptographic Protocols: Digital Signatures

Adding a *digital signature* to a message is a way of ensuring the recipient that the message came from the purported sender.

- Suppose that Alice's RSA public key is $(n,e)$ and her private key is $d$. Alice encrypts a plain text message $x$ using $E_{(n,e)}(x) = x^d$ **mod** $n$. She decrypts a ciphertext message $y$ using $D_{(n,e)}(y) = y^d$ **mod** $n$.
- Alice wants to send a message $M$ so that everyone who receives the message knows that it came from her.
  1. She translates the message to numerical equivalents and splits into blocks, just as in RSA encryption.
  2. She then applies her decryption function $D_{(n,e)}$ to the blocks and sends the results to all intended recipients.
  3. The recipients apply Alice's encryption function and the result is the original plain text since $E_{(n,e)}(D_{(n,e)}(x)) = x$.

Everyone who receives the message can then be certain that it came from Alice.

# Cryptographic Protocols: Digital Signatures

**Example**: Suppose Alice's RSA cryptosystem is the same as in the earlier example with key(2537,13), $2537 = 43 \cdot 59$, $p = 43$ and $q = 59$ are primes and $\gcd(e,(p-1)(q-1)) = \gcd(13, 42 \cdot 58) = 1$.

Her decryption key is d = 937.

She wants to send the message "MEET AT NOON" to her friends so that they can be certain that the message is from her.

**Solution**: Alice translates the message into blocks of digits 1204 0419 0019 1314 1413.

1. She then applies her decryption transformation $D_{(2537,13)}(x) = x^{937}$ **mod** 2537 to each block.
2. She finds (using her laptop, programming skills, and knowledge of discrete mathematics) that $1204^{937}$ **mod** $2537 = 817$, $419^{937}$ **mod** $2537 = 555$ , $19^{937}$ **mod** $2537 = 1310$, $1314^{937}$ **mod** $2537 = 2173$, and $1413^{937}$ **mod** $2537 = 1026$.
3. She sends 0817 0555 1310 2173 1026.

When one of her friends receive the message, they apply Alice's encryption transformation $E_{(2537,13)}$ to each block. They then obtain the original message which they translate back to English letters.